Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Сетевые информационные технологии и сервисы»

**ОТЧЁТ**

**о прохождении учебной (технологической) практики**

**на кафедре Сетевых информационных технологий и сервисов (СИТиС)**

Выполнил:

Студент группы БСТ2001

Кувшинов М.И.

Руководитель практики:

Спец. учебной лаборатории кафедры СИТиС Тришина С.В

Москва 2022

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ**

**по учебной практике**

Регулировка освещения помещения в зависимости от уровня освещения на улице.

**Цель учебной практики:**

Получение навыков построения схем Arduino, разработки программного обеспечения для данных схем и построения систем управления базы данных. Продемонстрировать результаты работы с помощью отчёта и презентации.

**Задачи учебной практики:**

* Составление ER диаграммы (схемы взаимосвязей сущностей и их атрибутов) предметной области.
* Определение составных частей системы и их взаимоувязка в единый макет.
* Изучение и инсталляция симуляторов для аппаратно-программных средств семейства Arduino.
* Составление схемы взаимодействия в выбранном симуляторе.
* Проектирование базы данных для предметной области.
* Проектирование и реализация графического интерфейса и взаимодействие с БД.
* Составление отчета и презентации по теме практики.

Содержание

[**1.** **Описание предметной области** 4](#_Toc107405723)

[**2.** **Выделение сущностей предметной области и атрибутов сущностей** 5](#_Toc107405724)

[**3.** **Составление ER-диаграммы сущностей предметной области** 6](#_Toc107405725)

[**4.** **Определение составных частей системы и их взаимоувязка в единый макет** 7](#_Toc107405726)

[**5.** **Изучение и инсталляция симуляторов для аппаратно-програмных средств семейства Arduino.** 8](#_Toc107405727)

[**6.** **Выбор электронных компонентов, необходимых для функционирования прототипа макета** 9](#_Toc107405728)

[**7.** **Составление схемы взаимодействия в выбранном симуляторе** 10](#_Toc107405729)

[**8.** **Проектирование базы данных для предметной области** 11](#_Toc107405730)

[**9.** **Проектирование и реализация графического интерфейса** 12](#_Toc107405731)

[**Список использованных источников** 14](#_Toc107405732)

[**ПРИЛОЖЕНИЕ А** 15](#_Toc107405733)

[**ПРИЛОЖЕНИЕ Б** 16](#_Toc107405734)

[**ПРИЛОЖЕНИЕ В** 17](#_Toc107405735)

1. **Описание предметной области**

Результатом данной работы должна быть система, которая автоматически регулирует освещение помещения, в зависимости от освещения улицы. Для реализации следует продумать использование датчиков, которые считывают показатели освещения улицы и передают данные для освещения помещения.

В итоге, должен получиться программный проект, который будет содержать программный интерфейс для взаимодействия с аппаратным комплексом, а так же базу данных проекта, которая будет содержать данные нашего программно-аппаратного комплекса.

1. **Выделение сущностей предметной области и атрибутов сущностей**

Для понимания будущей модели освещения помещения выделим необходимые атрибуты и их сущности:

1. Датчик освещения: показатель освещения улицы;
2. Источник света: свет, освещает помещение;
3. Помещение: освещается источником света, электроника.
4. Улица: освещение улицы;
5. **Составление ER-диаграммы сущностей предметной области**

На рисунке 3.1 изображена ER-диаграмма сущностей предметной области.

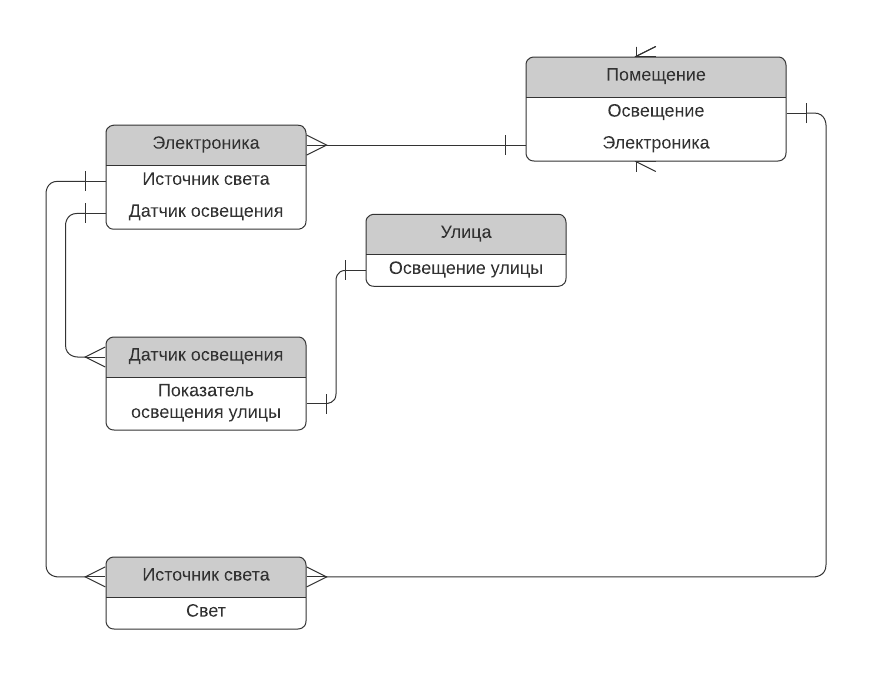


Рисунок 3.1 - ER-диаграмма

1. **Определение составных частей системы и их взаимоувязка в единый макет**

Для будущей системы мы планируем предполагаемый набор составных частей. Взаимоувяжем их в единый макет. Данный макет представлен на рисунке 4.1.

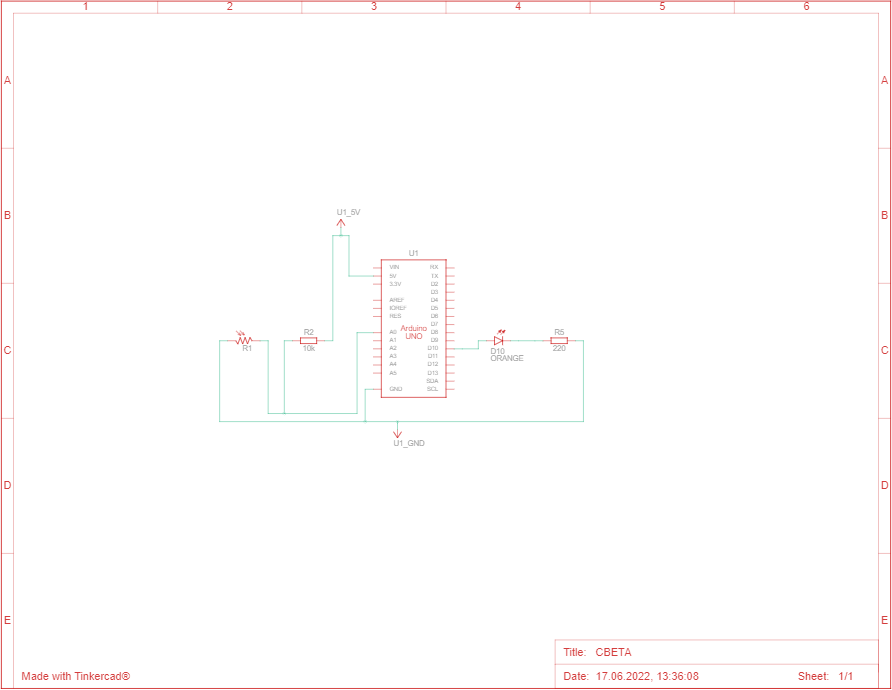


Рисунок 4.1 – макет схемы

1. **Изучение и инсталляция симуляторов для аппаратно-програмных средств семейства Arduino.**

В качестве эмулятора для Arduino был выбран Autodesk Tinkercard. Данный эмулятор является браузерным, поэтому он очень удобен и не требует инсталляции.

1. **Выбор электронных компонентов, необходимых для функционирования прототипа макета**

Будущий макет будет содержать в себе следующие составные части системы:

* Плата Arduino Uno R3;
* Фоторезистор;
* Оранжевый Светодиод;
* Резистор 1 кОм;
* Резистор 220 Ом;
* Макетная плата.

1. **Составление схемы взаимодействия в выбранном симуляторе**

Составив список компонентов, перейдём к составлению схемы в выбранном эмуляторе. Данная схема представлена на рисунке 7.1.

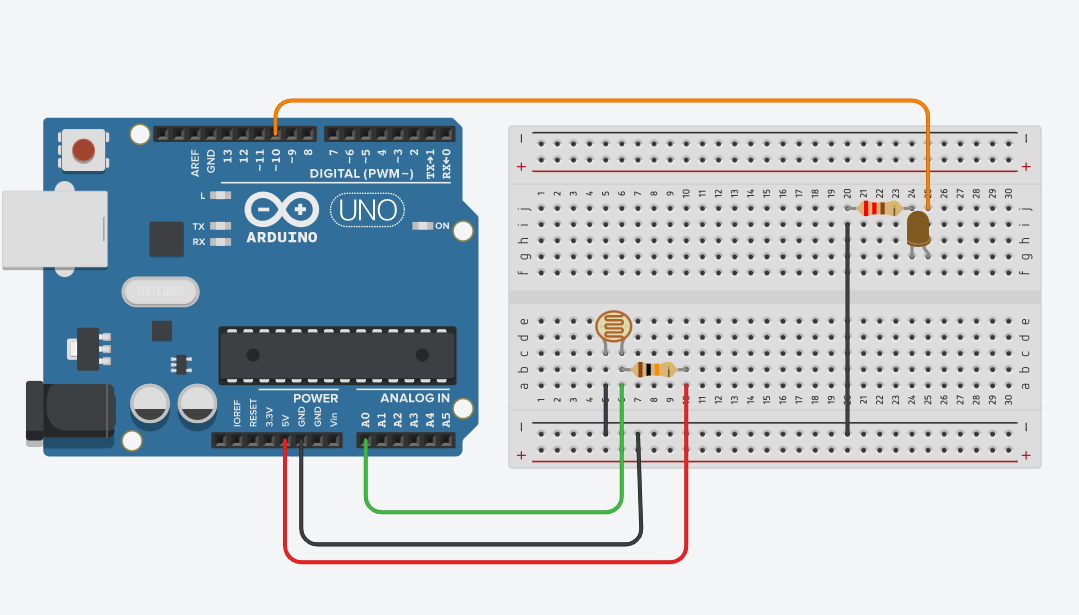


Рисунок 7.1 – схема

1. **Проектирование базы данных для предметной области**

База данных – неотъемлемая часть любого приложения. В данном проекте база данных реализована в эмуляторе Tinkercad с помощью библиотеки EEPROM. Данная библиотека использует динамическую память EEPROM, в которую мы будем записывать данные, показатели фоторезистора и светодиода. С помощью команды read “адресс” мы сможем просматривать эти данные. Запись происходит автоматически с помощью функции EEPROM.update(address, value), которая записывает значение (наши показатели) по определённому адресу, в нашем случае это 0 и 1.

На рисунке 8.1 представлен вывод из нашей БД. Вся кодовая состовляющая представлена в приложении А.



Рисунок 8.1 – база данных

1. **Проектирование и реализация графического интерфейса**

Для реализации интерфейса нашей программы используется среда разработки от компании Microsoft под названием Visual Studio 2022. Эта среда разработки позволяет нам с помощью фреймворка Windows Forms разработать свой программный интерфейс на языке программирования С#.

Создав проект, нам необходимо сделать запуск нашего аппаратно-технического проекта. С помощью элемента управления LinkLabel сделан запуск нашего проекта, а именно, переход по ссылке в эмулятор Tinkercad, в котором и реализована наша схема. С помощью трёх кнопок есть возможность просмотра ER-диаграммы, чертежа и самой схемы.

На рисунке 9.1 изображён вид реализуемого интерфейса.

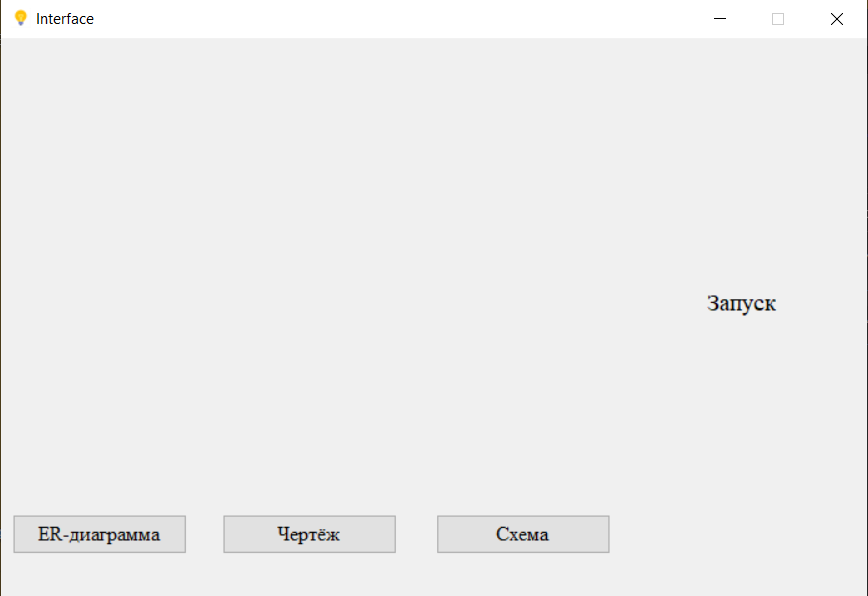


Рисунок 9.1 – интерфейс

В случае, если запуск происходит не на OC Windows, то для этого была разработана программа интерфейса на языке программирования Java, с помощью среды разработки IntelliJ IDEA и библиотеки swing, позволяющая разрабатывать кроссплатформенные приложения.

На рисунке 9.2 изображён вид реализуемого интерфейса.

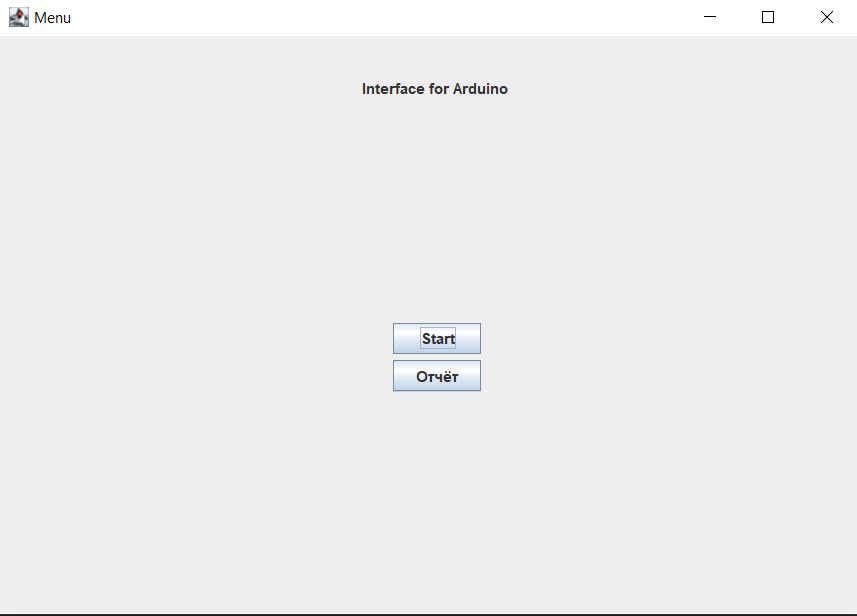


Рисунок 9.2 - интерфейс

**Список использованных источников**

1. ГОСТ 7.32-2017 СИБИД. Отчет о научно-исследовательской работе. Структура и правила оформления (с Поправками).[Правовой информационный ресурс]. – 2017. – URL: http://www.consultant.ru/document/cons\_doc\_LAW\_292293/(дата обращения 30.06.2022).
2. Autodesk Tinkercad.

[Программный эмулятор Arduino]. – 2022. –

URL: <https://www.tinkercad.com/dashboard>

(дата обращения 30.06.2022).

1. LucidApp.

[Сервис для создания диаграмм и блок-схем]. – 2022 –

URL: <https://lucid.app/documents#/dashboard>

(дата обращения 30.06.2022).

1. Metanit.

[Сайт о программировании]. – 2022 –

URL: https://metanit.com

(дата обращения 30.06.2022).

# **ПРИЛОЖЕНИЕ А**

Листинг Arduino

#include <EEPROM.h>

#define PIN\_LED 10

#define PIN\_PHOTO\_SENSOR A0

byte value;

int address;

String buffer;

void setup() {

Serial.begin(9600);

pinMode(PIN\_LED, OUTPUT);

Serial.println("Use the \*read \*0 or 1\* command to view the data");

}

void loop() {

buffer = Serial.readString(); //строка для БД

int val = analogRead (PIN\_PHOTO\_SENSOR); //вычисление фоторезистора

EEPROM.update(0, val); //запись в БД

int ledPower = map(val, 0, 500, 0, 255); //вычисление светодиода

EEPROM.update(1, ledPower); //запись в БД

analogWrite(PIN\_LED, ledPower);

if (buffer.startsWith("read")) { //вывод из БД

address = buffer.substring(5,6).toInt();

if (address == 0) {

Serial.print("Data from the Photosensor: ");

Serial.println(EEPROM.read(address));

} else if (address == 1) {

Serial.print("Data from the Diode: ");

Serial.println(EEPROM.read(address));

} else {

Serial.println("The address does not exist");

}

}

}

# **ПРИЛОЖЕНИЕ Б**

Листинг интерфейса С#

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace InterfaceForPraktika

{

public partial class Interface : Form

{

public Interface()

{

InitializeComponent();

MaximizeBox = false;

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

System.Diagnostics.Process.Start("https://www.tinkercad.com/things/krbligoUHrb");

}

private void button1\_Click(object sender, EventArgs e)

{

pictureBox1.Visible = true;

pictureBox2.Visible = false;

pictureBox3.Visible = false;

}

private void button2\_Click(object sender, EventArgs e)

{

pictureBox2.Visible = true;

pictureBox1.Visible = false;

pictureBox3.Visible = false;

}

private void button3\_Click(object sender, EventArgs e)

{

pictureBox3.Visible = true;

pictureBox1.Visible = false;

pictureBox2.Visible = false;

}

}

}

# **ПРИЛОЖЕНИЕ В**

Листинг интерфейса Java

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
import java.io.File;  
import java.io.IOException;  
import java.net.URL;  
  
public class Main {  
 public static void main(String[] args) {  
 JFrame window = new JFrame("Menu");  
 Toolkit toolkit = Toolkit.*getDefaultToolkit*();  
 Dimension dimension = toolkit.getScreenSize();  
 window.setBounds(dimension.width/2 - 300, dimension.height/2 - 300, 700, 500);  
 window.setVisible(true);  
 window.setDefaultCloseOperation(WindowConstants.*EXIT\_ON\_CLOSE*);  
 window.setLayout(null);  
 JLabel text = new JLabel("Interface for Arduino");  
 text.setBounds(290,30, 150, 25);  
 window.add(text);  
 JButton start = new JButton("Start");  
 start.setBounds(315,230,70,25);  
 JButton doc = new JButton("Отчёт");  
 doc.setBounds(315,260,70,25);  
 window.add(start);  
 window.add(doc);  
 ActionListener listener1 = new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 *openWebpage*("https://www.tinkercad.com/things/krbligoUHrb");  
 }  
 };  
 start.addActionListener(listener1);  
 ActionListener listener2 = new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 try {  
 if (Desktop.*isDesktopSupported*()) {  
 Desktop.*getDesktop*().open(new File("src/Awesome.docx"));  
 }  
 } catch (IOException ioe) {  
 ioe.printStackTrace();  
 }  
 }  
 };  
 doc.addActionListener(listener2);  
 }  
 public static void openWebpage(String urlString) {  
 try {  
 Desktop.*getDesktop*().browse(new URL(urlString).toURI());  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}